# A Linear-System-Theoretic View of Discrete-Event Processes and Its Use for Performance Evaluation in Manufacturing

GUY COHEN, DIDIER DUBOIS, JEAN PIERRE QUADRAT, AND MICHEL VIOT

*Abstract*—**A discrete-event system is a system whose behavior can be described by means of a set of time-consuming activities, performed according to a prescribed ordering. Events correspond to starting or ending some activity. An analogy between linear systems and a class of discrete-event systems is developed. Following this analogy, such discrete-event systems can be viewed as linear, in the sense of an appropriate algebra. The periodical behavior of closed discrete-event systems, i.e., involving a set of repeatedly performed activities, can be totally characterized by solving an eigenvalue and eigenvector equation in this algebra. This problem is numerically solved by an efficient algorithm which basically consists of finding the shortest paths from one node to all other nodes in a graph. The potentiality of this approach for the performance evaluation of flexible manufacturing systems is emphasized; the case of a flowshop-like production process is analyzed in detail.**

## Introduction

LINEARITY is a most welcome property of systems since it significantly simplifies their analysis. However, most systems encountered in applications are nonlinear. Generally, coping with nonlinearity consists of considering a system as locally linear under prescribed behavior conditions; but linearization is not the only thinkable approach for some types of systems, for instance, discrete-event systems.

The name "discrete-event dynamical systems" is by now widely known to designate systems whose behavior can be completely characterized by the knowledge of starting and ending times of activities. The structure of the set of activities can be expressed by timed event-graphs [3]. The equations describing the behavior of the system usually require the maximum or the minimum of several quantities to be calculated. A typical example of a discrete-event system is a network of waiting lines. Many discrete production processes can be described as discrete-event systems. These systems are those considered by discrete-event simulation languages [14].

The purpose of this paper is first to indicate that discrete-event systems, when deterministic, are linear in the sense of an unusual algebra already known in the literature under names such as "path-algebra" [4], "minimax algebra" [8], [9], or "dioid" [16], [18]. A state-space representation of discrete-event systems can then be established. It is then possible to efficiently calculate the periodical steady state of *closed* discrete-event systems, i.e., involving a finite set of activities repeatedly performed by means of a limited amount of available resources. Discrete-event systems can also be represented by timed Petri nets [28] whose transitions are the activities, and tokens are resources [12]. Hence, our approach is also relevant for performance evaluation of distributed computer systems (Ramamoorthy and Ho [27]) as well as parallel computation models (Reiter [29]). Our attempt is distinct from that of Ho and Cassandras [22], since this paper aims at studying the asymptotical behavior of discrete-event systems, while Ho and Cassandras are interested in perturbation analysis.

Our motivation is rather the modeling of production processes such as those involved in recently emerged automated production systems known as "flexible manufacturing systems." For technical details on these systems see Groover [19] and the proceedings of the FMS-1 conference [15]. Thus far, flexible manufacturing systems have been modeled by stochastic networks of queues (Solberg [32], Hildebrant [20], Secco-Suardo [30]). These models are useful to predict long range performance of FMS's when very little information is available about the actual system. Most of FMS queuing models are however very much approximate, basically because they assume exponentially distributed processing times on machines in order to obtain computationally attractive models. However, in spite of slight variations in processing times, flexible manufacturing systems are basically deterministic between two major breakdowns. As a consquence FMS models based on Jackson networks of queues [23] do not provide quantitatively reliable performance prediction. Although approximate methods can cope with nonclassical networks of queues and give more accurate results (e.g., Buzacott and Shanthikumar [31] for open systems, Cavaillé and Dubois [5] for closed systems), still, no insight is provided about the way the FMS should be controlled in order to achieve some predicted performance.

However, it is worth mentioning that some progress along this line may come from a nonstochastic approach of queuing networks called "operational analysis" (Denning and Buzen [11]). Basically, operational analysis establishes relationships between values of relevant quantities observed from simulation runs. These relationships turn out to be exactly the ones obtained in Jackson networks under strong stochastic assumptions, but the "operational" framework is much less drastic. Suri [33] has explained the robustness of the Jacksonian networks, using this framework, and in the scope of production systems analysis. Recently, Dallery and David [10] have proposed a new performance evaluation algorithm, based on operational analysis, but obviating the need for simulation runs.

The approach presented in this paper not only enables accurate performance evaluation, but also suggests simple real time control rules under which the flexible manufacturing system has a regular and stable behavior when no major breakdown occurs, the production requirements are satisfied, and the bottleneck machines are fully utilized. Dealing with major breakdowns and production reconfiguration is supposed to be the task of an upper control level in a hierarchy such as the one described in Kimemia *et al.* [25], for instance. It is claimed that our paper provides some justification to this hierarchical control scheme where short-term

Manuscript received June 7, 1983; revised February 6, 1984. Paper recommended by P. R. Kumar, Past Chairman of the Stochastic Control Committee.

G. Cohen is with the Centre d'Automatique et d'Informatique, ENSMP, Fontainebleau, France.

D. Dubois was with the Department d'Automatique (DERA), CERT, Toulouse, France. He is now with L.S.I., P. Sabatier University, Toulouse, France.

J. P. Quadrat is with the Institut de Recherche en Informatique et Automatique, Le Chesnay, France.

M. Viot is with the Ecole Polytechnique, Palaiseau, France.

deterministic discrete aspects are considered at the lower level, while the long term stochastic, continuous flow aspects are dealt with at the upper level. However, this topic is beyond the scope of this study.

Our approach is basically adapted to the modeling of repetitive production processes, such as those encountered in some types of flexible manufacturing systems dedicated to medium-volume production. One obvious limitation of our work is the assumption of a decision-free system, i.e., processing sequences as well as part sequences at machines must be fixed. However, once these requirements have been satisfied, a complete characterization of the periodical behavior of the system can be simply calculated. This steady state is reached after a finite number of steps.

This analysis can be viewed as a necessary step before design and control problems pertaining to the concerned class of manufacturing systems can be solved. Our approach extends and plays a role similar to that of critical path methods (CPM) in classical scheduling and sequencing problems.

This paper is organized as follows. The first section proposes a state-space representation of discrete-event systems, and derives a set of recurrent equations describing the behavior of closed systems. Section II provides mathematical results necessary to solve the recurrent equations; a complete characterization of the discrete-event system behavior can then be obtained regarding its steady-state behavior. Section III applies these results to the analysis of a flexible manufacturing system. The conclusion mentions the potential of this algebraic approach to encompass more general production processes. It is worth noticing that this type of approach to the analysis of production processes was already contemplated by Cuninghame-Green [7] a long time ago!

## I. A Linear Representation of Discrete-Event Systems

### A. Finite Deterministic Discrete-Event Systems

A finite deterministic discrete-event system can be viewed as a finite set of activities $\mathcal{A}$ and a finite set of resources $\mathcal{R}$ shared by activities. The order in which activities are performed can be described by an acyclic oriented graph $(\mathcal{A}, \mathcal{U})$ [2]. For any two activities $a_i$ and $a_j$, $(i, j) \in \mathcal{U}$ means activity $a_i$ precedes activity $a_j$. The graph is assumed to be connected (see Fig. 1). With each resource $r \in \mathcal{R}$ is associated a unique path $P_r$ in the graph, i.e., a sequence of consecutive activities in the graph. The set of paths $\mathcal{P} = \{P_r | r \in \mathcal{R}\}$ is supposed to be a coverage of $(\mathcal{A}, \mathcal{U})$, i.e.,

$$\forall (i, j) \in \mathcal{U}, \ \exists r, \qquad (ij) \in P_r.$$

Indeed one reason why $a_i$ precedes $a_j$ may be that they share some common resource and cannot use it at the same time. Some decision has been made regarding the order in which activities are visited by each resource. However, there may also be a precedence constraint specifying that $a_i$ must precede $a_j$ for a proper achievement of the process described by $(\mathcal{A}, \mathcal{U})$.

*NB:* When there are two activities requiring the same resource, and we do not know which of these activities must use the resource first, the event-graph is then said to be disjunctive (e.g., [1, ch. 12]). The system is then not decision-free, hence is not deterministic. A deterministic discrete-event system is defined by the 4-uple $(\mathcal{A}, \mathcal{U}, \mathcal{R}, \mathcal{P})$.

*Example:* A production process can be described by the processing of $n$ workpieces in a workshop containing $m$ machines. Each workpiece has a processing sequence which consists of a finite sequence of machines which are visited according to the order defined by the sequence. To simulate the system, the sequencing of the tasks on each machine may be prescribed. Hence,

$\mathcal{A}$ = set of processing tasks for producing the $n$ workpieces; they can be denoted by pairs $(i, j)$ which mean part $j$ on machine $i$;

$\mathcal{R}$ = set of machines and workpieces;

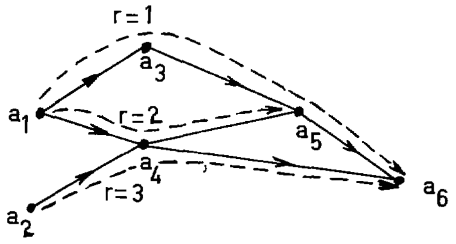


Fig. 1.

$\mathcal{U}$ = set of precedence constraints between tasks, due to the processing sequences of workpieces or the sequencing of workpieces on machines;

$\mathcal{P}$ = the processing sequences and the workpiece sequences on machines.

Any resource trajectory $P_r$ contains an initial activity and a final activity, respectively, denoted by $a_1(r)$ and $a_{k_r}(r)$. The sets of initial and final activities are then, respectively, defined by

$$\mathcal{I} = \{a_1(r) | r \in \mathcal{R}\}$$

$$\mathcal{F} = \{a_{k_r}(r) | r \in \mathcal{R}\}.$$

Notice that $\mathcal{I}$ (respectively, $\mathcal{F}$) contains all activities without predecessors (respectively, successors) at least.

In the example of Fig. 1

$$P_1 = a_1 a_3 a_5 a_6; \ P_2 = a_1 a_4 a_5;$$

$$P_3 = a_2 a_4 a_6; \ \mathcal{I} = \{a_1, a_2\}; \ \mathcal{F} = \{a_5, a_6\}.$$

Each arc $(i, j)$ of $\mathcal{U}$ is weighted by a real positive number $t_{ij}$ which can be viewed as the sum of the duration of activity $i$, say $t_i$, and the time required to switch from activity $i$ to activity $j$ (e.g., transport time, setup time), say $\delta_{ij}$. The *weight* of a path in $(\mathcal{A}, \mathcal{U})$ is the sum of the weights of the arcs contained in the path. Its *length* is the number of its arcs.

### B. Algebraic Representation

Let $x_i$ be the earliest starting time of activity $a_i$, and $u_r$ be the time when resource $r$ is available for the first activity of its trajectory $P_r$.

If $(j, i) \in \mathcal{U}$, then $x_i \geq x_j + t_{ji}$. If $i$ is the first activity for resource $r$, then $x_i \geq u_r$.

Let $\Gamma^-(i)$ be the set of predecessors of activity $i$ and $R^o(i)$ be the set of resources such that $a_1(r) = i$; then $x_i$ is clearly defined by

$$\forall a_i \in \mathcal{A}, \ x_i = \max \left( \max_{j \in \Gamma^-(i)} x_j + t_{ji}, \ \max_{r \in R^o(i)} u_r \right). \tag{1}$$

Note that this equation is a dynamic programming one, up to the $u_r$ terms. Let $A$ be the weighted incidence matrix of $(\mathcal{A}, \mathcal{U})$ defined by its entries

$$\begin{aligned} A_{ij} &= t_{ij} && \text{if } (i, j) \in \mathcal{U} \\ &= -\infty && \text{if } (i, j) \notin \mathcal{U}. \end{aligned}$$

$A$ is an $N \times N$ matrix with $N = |\mathcal{A}|$. Let $B$ be an $R \times N$ matrix, where $R = |\mathcal{R}|$ such that:

$$\begin{aligned} b_{ri} &= 0 && \text{if } a_1(r) = i \\ &= -\infty && \text{otherwise.} \end{aligned}$$

$B$ clearly indicates which are the starting activities for all resources. Then provided the following conventions are adopted (Gondran-Minoux [13])

$$\forall a, b \in \mathbb{R} \qquad a \oplus b \triangleq \max(a, b) \tag{2}$$

$$a \cdot b \triangleq a+b \tag{3}$$

for all matrices $A$, $A'$,

$$(A \oplus A')_{ij}=A_{ij} \oplus A'_{ij} \triangleq \max (A_{ij}, A'_{ij}) \tag{4}$$

$$(A \cdot A')_{ij} \triangleq \max_k A_{ik}+A'_{kj}. \tag{5}$$

(In (4) both $A$ and $A'$ have identical size; in (5) the number of rows in $A$ equals the number of columns in $A'$, as in usual linear algebra.) Equation (1) can be more compactly written as

$$X=XA \oplus UB \tag{6}$$

where $X = (x_1 \cdots x_N)$, $U = (u_1 \cdots u_R)$, and $XA$ is short for $X \cdot A$.

Now let $Y$ be the vector of earliest times $y_r$ when resources are released from their activities. If $C$ is an $N \times R$ matrix whose entries are, where appropriate, the durations of the last tasks performed by activities, i.e.,

$$C_{ir}=t_i \quad \text{if } a_{k_r}(r)=i \text{ for some } r$$

$$= -\infty \quad \text{otherwise.}$$

Then $Y$ is easily obtained from $X$ by

$$Y=XC. \tag{7}$$

Note that (6) and (7) are similar to the state-space representation of linear systems. However, this is not just a matter of notation. The analogy goes deeper as indicated in the subsequent sections.

Before proceeding to the closed-loop representation, let us solve the set of equations (6) and (7). It requires a lemma which clarifies the structure of the set of real matrices under operations (4) and (5).

*Lemma 1:* $\oplus = \max$ is commutative, associative over $\mathbb{R}$; $\epsilon = -\infty$ is the identity such that $\epsilon \oplus a = a$, $\forall a \in \mathbb{R}$. Moreover, $a \oplus a = a$ (idempotency). The operation $\cdot$ = addition is distributive over $\oplus$. $\epsilon$ is an absorbing element for $\cdot$, i.e., $\epsilon \cdot a = \epsilon$, $\forall a \in \mathbb{R}$. $e = 0$ is the identity of $\cdot$.

The matrix product defined by (5) is associative and distributive over the matrix addition. Moreover, matrix addition is idempotent.

*Theorem 1:* The equation $X = XA \oplus UB$ has a unique solution once $U$ is known; and it is $X = UBA^*$.

*Proof:* (See, e.g., Gondran-Minoux [13] and Shier [34].) Let $X$ be a solution, then

$$X=(XA \oplus UB)A \oplus UB$$
$$=XA^2 \oplus UB(E \oplus A) \text{ from Lemma 1.}$$

Here $E$ is the identity of the matrix product, i.e., $E_{ii} = e = 0$ and $E_{ij} = \epsilon = -\infty$ if $i \neq j$. Iterating $N - 1$ times

$$X=XA^N \oplus UB(E \oplus A \oplus A^2 \oplus \cdot \oplus A^{N-1}) \tag{8}$$

where $A^n = A^{n-1}, A = AA^{n-1}, n = 2, N - 1$.

Now entry $A^n_{ij}$ of $A^n$ clearly contains the maximal weight of a path from $i$ to $j$ with $n$ arcs exactly. Since $A$ is the incidence-matrix of an acyclic graph any path has at most $N - 1$ arcs. Hence, $A^N = \phi$, the null matrix whose entries are $\epsilon$. Hence, any solution of (6) reads

$$X=UB(E \oplus A \oplus A^2 \oplus \cdots \oplus A^{N-1}) \triangleq UBA^*. \tag{9}$$

$A^*$ contains as entries the maximal weights of paths between two nodes and is very similar to a resolvent in usual linear algebra. Q.E.D.

Note that $A^*$ can be simply obtained by any longest path algorithm, as in CPM methods.

## C. Dynamic Representation of Closed Discrete-Event Systems

A discrete-event system is said to be closed if it is a finite deterministic discrete-event system with feedback from the final activities to the initial ones. The feedback effect stems from the use of the finite set of resources to repeatedly perform the activities. Formally, a closed discrete-event system is a 5-uple ($\mathcal{A}$, $\mathcal{U}$, $\mathcal{R}$, $\mathcal{P}$, $\mathcal{B}$) where the first four symbols have the same meaning as in Section I-A and $\mathcal{B}$ is a bipartite graph ($\mathcal{F}$, $\mathcal{I}$, $L$) where $\mathcal{F}$ (respectively, $\mathcal{I}$) is the set of final (respectively, initial) activities in $\mathcal{A}$ and $L \subseteq \mathcal{F} \times \mathcal{I}$ contains the arcs of $\mathcal{B}$

$$(i, j) \in L \Leftrightarrow \exists r \in \mathcal{R}, \quad i=a_{k_r}(r), j=a_1(r).$$

$L$ simply expresses the fact that once a resource has completed its last activity, then it starts again with the first.

Let $X(n)$ be the vector of earliest starting times of activities in $\mathcal{A}$ for the $n$th time, and $U(n)$ the corresponding starting times for resources. Any resource $r$ can start being used by the first activity for the $n$th time provided it has been released by the last activity for the $(n - 1)$th time, and the time necessary to switch from one activity to the other has elapsed.

This fact leads to the following relationship:

$$U(n)=Y(n-1)K \tag{10}$$

where $K$ is an $R \times R$ matrix such that $K_{rs} = \epsilon$, $\forall r \neq s$ and $K_{rr}$ = time to switch from activity $a_{k_r}(r)$ to $a_1(r)$ for resource $r$. Through $K_{rr}$ it is possible to create delays in the starting times of activities. Now using (7), (9), and (10), the $n$th earliest release dates of resources can be expressed in terms of their $n$ - 1th earliest release dates by

$$Y(n)=U(n)BA^*C$$
$$=Y(n-1)KBA^*C. \tag{11}$$

Hence, the knowledge of initial availability dates $U(1)$ for resources enables one to derive in a recursive way the sequence $Y(1)$, $Y(2) \cdots Y(n) \cdots$ from which all of the information ($X(1)$, $X(2) \cdots X(n) \cdots$) can be retrieved. Equation (11) is a forward dynamic programming equation carried over to an infinite number of steps.

Now, instead of assuming that each resource $r$ is available after time $u_r$, we consider it can be used only before a prescribed date $v_r$.

## D. The Dual System

Let $q_i$ denote the *latest* starting time of activity $i$ of the finite deterministic discrete-event system considered in Section I-A, $v_r$ denotes the latest time when resource $r$ *must* be released once activities have been performed, and $z_r$ the latest time when resource $r$ must be available so that activities can be performed in due time, i.e., so that resource $r$ will indeed be released before time $v_r$, for all $r$.

Clearly, if $(j, i) \in U$, then $q_j \leq q_i - t_{ji}$ if $a_i$ is the last activity for resource $r$, then $q_i \leq v_r - t_i$. If $\Gamma^+(i)$ is the set of successors of activities $a_i$ and $R^\infty(i)$ is the set of resources with final activity $i$, we clearly have

$$\forall a_i \in \mathcal{R}, \; q_i=\min \Big( \min_{j \in \Gamma^+(i)} q_j-t_{ji}, \; \min_{r \in R^\infty(i)} v_r-t_i \Big). \tag{12}$$

Hence, by denoting $Q$, $V$, $Z$ the vectors such that $Q_i = -q_i$, $V_r = -v_r$, $Z_r = -z_r$, the following matrix equations hold:

$$Q=AQ \oplus CV \tag{13}$$

$$Z=BQ. \tag{14}$$

Equations (13), (14) are clearly the dual system of (6), (7). The solution is

$$Z = BA^*CV.$$

The closed form is obtained by noticing that

$$V(n) = KZ(n+1), \tag{15}$$

i.e., the latest release dates for resources on activities of set $n$ are conditioned by the latest admissible availability dates on the following set of activities.

Hence, the following recursive equation, dual of (11), is obtained:

$$V(n) = KBA^*CV(n+1). \tag{16}$$

The solution of (16) assumes the knowledge of due dates when resources must be released, i.e., a "final state" under the form of a vector $V(n^*)$ for some $n^*$.

## II. Solving (max, +)-Recurrence Equations

In this section we consider the asymptotical behavior of recurrent equations of the form $Y(n) = Y(n-1)M$ where $M$ is a real matrix and the matrix product is in the sense of (5). A basic issue is the eigenvalue problem $\lambda Y = YM$. Extensive results are provided but proofs are only outlined most of the time. A complete study of this problem appears in [6]; the book by Cuninghame-Green [9] also contains relevant material. This section contains the mathematical and algorithmic background of the approach, and its results are applied to the analysis of production systems in the next section.

### *A. Eigenvalues*

Let $G(M) = (\mathfrak{N}(M), \mathfrak{U}(M))$ be the graph whose weighted incidence matrix is $M$. Namely

$$M_{ij} > \epsilon \Leftrightarrow (i, j) \in \mathfrak{U}(M) \tag{17}$$

$M$ has dimension $N \times N$.

In the following it is assumed that $G(M)$ is strongly connected [2], i.e., from any node to any other, there is a path. $M$ is then called an irreducible matrix. Note that the $n$th power of an irreducible matrix, in the sense of matrix product (5) may not be irreducible. A vector $y \neq \epsilon$ is said to be an eigenvector if there is a real number $\lambda$ called eigenvalue, such that $yM = \lambda y$.

*Lemma 2:* If $\lambda y = yM$, then $\lambda > \epsilon$ and $y_i > \epsilon$, $\forall i = 1, N$.

*Proof:* This result is easily obtained by showing that $\lambda = \epsilon$ or $y_i = \epsilon$ for some $i$, are not compatible with the irreducibility of $M$ (see [6]). Q.E.D.

Given a path $P = i_1, \cdots i_k$ in $G(M)$, the length $l(P)$ and the weight $w(P)$ are defined, respectively, by

$$l(P) = k - 1 = \text{number of arcs on } P$$

$$w(P) = M_{i_1, i_2} \cdot M_{i_2 i_3} \cdots M_{i_{k-1} i_k}.$$

Note that $W(P)$ is actually obtained by a sum in the usual sense.

If path $P$ is such that $i_1 = i_k$, it is called a circuit, and denoted $\gamma$. We can now define, for any circuit $\gamma$ in $G(M)$ its average weight $\bar{w}(\gamma) = w(\gamma)^{1/l(\gamma)}$, which should be understood as $w(\gamma)/l(\gamma)$ in the usual algebra. Let $w = \Sigma_\gamma\, w(\gamma)^{1/l(\gamma)}$ be the maximal average weight of circuits in $G(M)$. Since $G(M)$ has a finite number of nodes, only circuits such that $l(\gamma) \leq N$ need be considered in the summation, and thus, $w$ exists, and there is at least one circuit $\gamma$ such that $\bar{w}(\gamma) = w$. $\gamma$ is then called a critical circuit.

Lastly, let $M^*$ and $M^+$ be defined, whenever they exist, by

$$M^* = E \oplus M \oplus M^2 \oplus \cdots \oplus M^n \oplus \cdots \tag{18}$$

$$M^+ = M \oplus M^2 \oplus \cdots \oplus M^n \oplus \cdots. \tag{19}$$

From [5, ch. 3, sect. 2.3, Theorem 1] $M^*$ and $M^+$ exist only if every circuit in $G(M)$ satisfies $w(\gamma) \leq e(\triangleq 0)$. In this case,

$$M^+ = \sum_{i=1}^{N-1} M^i;\ M^* = E \oplus M^+;\ M^+ = M \cdot M^* \tag{20}$$

and

$$M^+_{ij} = M^*_{ij}, i \neq j;\ M^+_{ii} = M^*_{ii} \Leftrightarrow M^+_{ii} = e.$$

If $w$ is the maximal average weight of circuits in $G(M)$, and if $M_w \triangleq w^{-1}M$ is the matrix deduced from $M$ by subtracting $w$ from each of its entries, then the maximal average weight of circuits in $G(M_w)$ is clearly $e$, so that $M^+_w$ and $M^*_w$ always exist.

*Theorem 2:* $\lambda = w$ is the unique eigenvalue of $M$. If $\gamma_o$ is a *critical* circuit then $\forall\ i \in \gamma_o$, row $i$ of $M^+_\lambda \triangleq \lambda^{-1}M^+$ is an eigenvector.

*Proof:* Since $\gamma_o$ is a critical circuit in $G(M)$, its weight in $G(M_w)$ is $e$, and it is a maximal weight circuit in $G(M_w)$. Hence, $(M^+_w)_{ii} = e\ \forall\ i \in \gamma_o$, and so rows $i$ of $M^+_w$ and $M^*_w$ are the same. Hence, $\forall i \in \gamma_o$, if $y = (M^+_w)_{i.}$, then

$$yM = w(M^+_w M_w)_{i.} = w(M^*_w M_w)_{i.} = w(M^+_w)_{i.} = wy.$$

Now if $\lambda$ is an eigenvalue with eigenvector $y$, let $\gamma$ be a circuit $i_1 - i_2 \cdots i_k - i_1$, then $\lambda y = yM$ implies

$$\lambda y_{i_2} \geq y_{i_1} \cdot M_{i_1 i_2};\ \lambda y_{i_3} \geq y_{i_2} \cdot M_{i_2 i_3};\ \cdots;\ \lambda y_{i_1} \geq y_{i_k} \cdot M_{i_k i_1}. \tag{21}$$

Now, since $y_{ik} > \epsilon$ (Lemma 2), and using substitution, (21) implies $\lambda^{l(\gamma)} \geq w(\gamma)$. Hence, $\forall \gamma$, $\lambda \geq \bar{w}(\gamma)$, i.e., $\lambda \geq w$.

Now for all $i = 1, N$, let $j(i)$ be such that $\lambda y_i = y_{j(i)} M_{j(i)i}$, where $A_{j(i)i}$ denotes the entry $(j(i), i)$ in $A$.

Let $H$ be the graph $(\mathfrak{N}(M), \mathfrak{U}')$ where

$$\mathfrak{U}' = \{(i, j(i)) \mid i = 1, N\} \subseteq \mathfrak{U}(M).$$

It is easy to figure out that since $H$ has as many nodes as arcs, and from any node starts exactly one a $H$ is a collection of $K \geq 1$ disjoint subgraphs each containing only one circuit $\gamma_k$. On every circuit $\gamma_k$ of $H$, (21) holds with equality, hence $\lambda = w(\gamma_k)^{1/l(\gamma_k)}$. But since $\lambda \geq w$, then $w(\gamma_k)^{1/l(\gamma_k)} = w = \lambda$. Q.E.D.

Note that $\forall k$, $\gamma_k$ is a critical circuit. The oriented graph $G_o(M) = (\mathfrak{N}_o, \mathfrak{U}_o)$ defined by

$$\mathfrak{N}_o = \{i \in \mathfrak{N}(M_w) \mid i \text{ belongs to a critical circuit}\}$$

$$\mathfrak{U}_o = \{(i, j) \in \mathfrak{N}(M_w) \mid (i, j) \text{ belongs to a critical circuit}\}$$

is called the critical graph of $M$.

Theorem 2 provides a graph theoretical interpretation of the eigenvalue $\lambda$ and of some eigenvectors: to each node $i$ in $G_o(M)$ corresponds an eigenvector $y^i = (M^+_w)_{i.}$, and each entry $y^i_j$ contains the maximal weight of paths from $i \in \mathfrak{N}_o$ to $j$ in $G(M_w)$. Each path of maximal weight belongs to $H$.

### *B. Eigenvectors*

We assume here $M$ has eigenvalue $e(=0)$ (otherwise we work on $M_w$). It is now indicated that eigenvectors obtained from $M^+$ contain a basis for the set of all eigenvectors.

*Lemma 3:* Any eigenvector $y$ is a linear combination of some rows in $M^+$. More specifically,

$$y = \sum_{i \in \mathfrak{N}_o} y_i M^+_{i}.$$

*Proof (Outline):* If $y$ is an eigenvector of $M$, then it is also an eigenvector of $M^n$ and since $\oplus$ is idempotent, it is an eigenvector of $M^+$, i.e.,

$$y = \sum_{i \in \mathfrak{N}(M)} y_i M_i^+.$$

Let $j \notin \mathfrak{N}_o$ and $i \in \mathfrak{N}_o$ such that $j$ and $i$ belong to the same connected component of graph $H$. Let $P$ be the path from $i$ to $j$ in $H$. Along this path inequalities (21) are equalities, and so $y_j = w(P)y_i$. Hence, if $j \notin \mathfrak{N}_o$

$$\forall l \;\; y_j M_{jl}^+ = y_i w(P) M_{jl}^+ \le y_i M_{il}^+.$$

This inequality stems from the fact that $M_{il}^+$ is the maximal weight of paths from $i$ to $l$.

*Theorem 3:* If the critical graph has only one connected component, then all rows $M_{i}^+$ with $i \in \mathfrak{N}_o$ are colinear.

*Proof:* Notice that $M_{i.}^+ \ge M_{ij}^+ M_{j.}^+$ holds since maximal weight paths from $i$ to another node do not cross $j$ usually. But if $i$ and $j$ are on the critical graph, then $M_{ij}^+ M_{ji}^+ = M_{ii}^+ = e$. Hence, if $i \in \gamma_o$, $j \in \gamma_o$

$$M_{ij}^+ \cdot M_{j.}^+ \le M_{i.}^+ \le M_{ij}^+ M_{ji}^+ M_{i.}^+ \le M_{ij}^+ M_{j}^+.$$

Hence, $M_{i.}^+ = M_{ij}^+ M_{j.}^+$. Q.E.D.

Using Lemma 3 and Theorem 3, it is clear that if the critical path is unique, the set of eigenvectors is of dimension 1. In the general case, it is no longer true, and the following result is proved in [6].

*Theorem 4:* If the critical graph $G_o(M)$ has $K$ connected components $C_k$, then the set of eigenvectors is of dimension $K$. A basis of eigenvectors can be obtained as $\{M_{ik}^+ | k = 1, K\}$ where $i_k$ is arbitrarily selected in $C_k$, $\forall k = 1, K$.

*Proof (Outline):* If $i$ and $j$ belong to $C_k$, then the same reasoning as for Theorem 3 yields $M_{i.}^+ = M_{ij}^+ \cdot M_{j.}^+$. If $i \in C_k$ and $M_{i.}^+ = \Sigma_{j \in \mathfrak{N}_o - C_k} \mu_j M_{j.}^+$, then $\mu_j = M_{ij}^+$ from Lemma 3. It is easy to see that $i \in C_k$ and $j \in \mathfrak{N}_o - C_k$ would be on the same critical path for some $j$. This is impossible. Hence, the set $\{M_{i_k}^+ | i_k \in C_k, k = 1, K\}$ is linearly independent. Now from Lemma 3 any eigenvector $y$ is such that

$$y = \sum_{j \in \mathfrak{N}_o} y_j M_{j.}^+ = \sum_{k=1}^{K} \sum_{j \in C_k} y_j M_{j.}^+$$

$$= \sum_{k=1}^{K} \left( \sum_{j \in C_k} y_j M_{ji_k}^+ \right) M_{i_k}^+.$$

Hence, a basis of eigenvectors is obtained. Q.E.D.

Particularly, if the critical circuit is unique then all the eigenvectors are colinear.

## *C. Periodicity*

From now on, it is assumed that $\forall n > 0$ matrix $M^n$ is irreducible. A sufficient condition for this is that $G(M)$ contains a loop $(i, i) \in \mathfrak{U}$. In the application to manufacturing systems, this assumption is not restrictive as seen in Section III.

A matrix $M$ with eigenvalue $e$ is said to be order-$d$-periodical if and only if there is an integer $n_o$ such that

$$\forall n \ge n_o, M^{n+d} = M^n. \tag{22}$$

As a consequence if $M$ has eigenvalue $\lambda$ and $M_\lambda$ is order-$d$-periodical then $M^{n+d} = \lambda^d M^n \; \forall \; n \ge n_o$.

In the following $M$ has eigenvalue $e$, and the critical path is $\gamma_o$, supposedly unique.

*Lemma 4:* If $\gamma_o$ is of length $d$ then $M^d$ has exactly $d$ critical paths which are loops around nodes $i \in \gamma_o$.

*Proof:* If $i \in \gamma_o$, then $(M^d)_{ii} = e$, hence loop $(i, i)$ is critical $\forall i \in \gamma_o$.

No other circuit of $G(M^d)$ is critical. Indeed any maximal weight path of length $k$ in $G(M^d)$ corresponds to at least one maximal weight path of length $kd$ in $G(M)$, both having the same weight. Hence, if $\gamma$ is critical in $G(M^d)$, then there is a critical circuit $\gamma_d$ in $G(M)$ whose length is $l(\gamma_d) = dl(\gamma)$. Hence, $l(\gamma) = 1$. Now a loop around $i \notin \gamma_o$ cannot be critical in $G(M^d)$ since there would be a critical circuit crossing $i$ in $G(M)$. Q.E.D.

*Theorem 5:* If the unique critical path $\gamma_o$ is of length $d$, then $M$ is order-$d$-periodical.

*Proof:* Let $i$ and $j$ be such that a maximal weight path from $i$ to $j$ in $G(M^d)$ crosses some $i_o \in \gamma_o$, and is of length $p \le N - 1$. Then

$$(M^d)_{ij}^+ = (M^{dp})_{ij} = (M^d)_{ii_o}^+ (M^d)_{i_oj}^+ \tag{23}$$

$\forall n \ge p$, $(M^{dd})_{ij} = (M^{pn})_{ij}$ since it is possible to loop $n - p$ times around $i_o$ without modifying the weight of the path.

Particularly, $\forall n \ge p$, $(M^{d(n+1)})_{i_o.} = (M^{dn})_{i_o.}$. Now assume $i \notin \gamma_o$ and $p_j < N$ is such that $(M^{dpj})_{ij} = (M^d)_{ij}^+$. Then,

$$\forall n \ge \max_j \; p_j, \forall i_o \in \gamma_o, (M^{dn})_{ij} \ge (M^{dp_{i_o}})_{ii_o} \cdot (M^{d(n-p_{i_o})})_{i_oj}, \forall j.$$

If $n$ is large enough $(M^{d(n-p_{i_o})})_{i_oj} = M_{i_oj}^+$, so that

$$\forall j \qquad (M^{dn})_{ij} \ge (M^{dp}i_o)_{ii_o} M_{i_oj}^+. \tag{24}$$

Let $\eta = \Sigma_{\gamma \ne \gamma_o} w(\gamma)^{1/l(\gamma)} < e$; $\eta$ is the second maximal average weights of circuits in $G(M^d)$. If no path of length $n$ in $G(M^d)$, and weight $(M^{dn})_{ij}$ crosses $i_o \in \gamma_o$, then it contains circuits of average weight less than or equal to $\eta$. Then, with $p_j < N$ such that $(M^{dpj})_{ij} = (M^d)_{ij}^+$

$$\forall j, (M^{dn})_{ij} \le (M^d)_{ij}^+ \eta^{n-p_j} \to \epsilon \qquad \text{when } n \to +\infty.$$

This is contradictory with (24). Hence, for some finite $n$, some maximal weight path from $i$ to $j$ in $G(M^d)$ crosses some $i_o \in \gamma_o$. So, $\forall i, j$

$$\exists i_o \in \gamma_o, \exists n_o(ij), \forall n \ge n_o(ij), (M^{dn})_{ij} = (M^d)_{ii_o}^+ (M^d)_{i_oj}^+ \tag{25}$$

for $n \ge \max_{ij} n_o(i, j)$, $M^d$ is order-1-periodical, and thus $M$ is order-$d$-periodical, i.e., $M^{dn} = M^{d(n+1)}$ Q.E.D.

More generally if $C_1 \cdots C_k$ are the critical components of $G_o(M)$, then the *order* of $C_k$ is defined by the g.c.d. of the (critical) circuit lengths in $C_k$.

1) If $G_o(M)$ has only one component of order $d$, then $M$ is order-$d$-periodical.

2) If $G_o(M)$ has $K$ components of order $d_k$, $k = 1, K$, then $M$ is order-$d$-periodical with $d = l.c.m(d_1 \cdots d_K)$-(least common multiple).

For a complete proof see [6].

Note that if $M^k$ is not irreducible, then $G(M^k)$ can be decomposed into disjoint strongly connected components, each corresponding to a distinct eigenvalue. Hence, the proof of Theorem 5, which underlies strong connectivity of $G(M^n)$ does not apply.

In the general case (25) reads

$$\exists n_o, \; n \ge n_o \Rightarrow (M^{dn})_{ij} = \sum_{i_o \in \mathfrak{N}_o} (M^d)_{ii_o}^+ (M^d)_{i_oj}^+. \tag{26}$$

## *D. Spectral Projectors*

Let $M$ be a matrix with eigenvalue $e$ and periodicity of order $d$. From Section II-C, the following quantities exist:

$$Q_i = \lim_{n \to +\infty} M^{dn+i} \qquad \text{for } i = 0, 1, \cdots d-1. \tag{27}$$

Clearly, $Q_i = M^i Q_o = Q_o M^i$.

Thus far, we only considered eigenvectors on the left. It is possible to consider eigenvectors on the right. They are of course colinear to any column $i$ of $M^+$ such that $i \in \mathfrak{N}_o$.

Let $\{\bar{y}_i = (M^d)^+_{i.}; i \in \mathfrak{N}_o\}$, $\{\bar{z}_i = (M^d)^+_{.i}; i \in \mathfrak{N}_o\}$ be bases of eigenvectors on the right and on the left, respectively, for $M^d$.

*Theorem 6:* $Q_o = \Sigma_{i \in \mathfrak{N}_o} \bar{z}_i \otimes \bar{y}_i$ where $\otimes$ is the external product in the (max, +) algebra (i.e., $(\bar{z}_i \otimes \bar{y}_i)_{jk} = (\bar{z}_i)_j \cdot (\bar{y}_i)_k$). $Q_o$ is a projector.

*Proof:*

$$(\lim_{n \to +\infty} M^{dn}) (\lim_{m \to +\infty} M^{dm}) = \lim_{n+m \to +\infty} M^{d(n+m)}.$$

Hence, $Q_o^2 = Q_o$. This property motivates for $Q_o$ the name "projector." Now from (26)

$$(Q_o)_{jl} = \sum_{i \in \mathfrak{N}_o} (M^d)^+_{ji} (M^d)^+_{il} = \sum_{i \in \gamma_o} (\bar{z}_i \otimes \bar{y}_i)_{jl}. \qquad \text{Q.E.D.}$$

If $M$ has $\lambda$ as its eigenvalue, then $\exists n_o$, $\forall n \geq n_o$

$$M^{dn+i} = Q_i \lambda^{dn} \qquad i = 0, 1, \cdots d-1.$$

$Q_o$ is called the spectral projector of $M^d$, and $Q_i$ is built from (27) applied on matrix $M\lambda^{-1}$. Clearly, $(Q_o)_{jl}$ is the maximal weight of paths from $j$ to $l$ in $G$ $[(M\lambda^{-1})^d]$, crossing the critical graph $G_o[(M\lambda^{-1})^d]$. Let $\{Y_n | n \in \mathbb{N}\}$ be a sequence of vectors defined by $Y_n = Y_{n-1}M$, and $Y_o$ is given; then:

$$\begin{aligned} Y_{nd} &= Y_o M^{nd} \\ &= Y_o Q_o \lambda^{(n-1)d} M^d \qquad \text{if } n \geq n_o \\ &= Y_o Q_o \lambda^{nd}. \end{aligned}$$

Hence, $Y_o Q_o M^d = Y_o Q_o \lambda^d$, i.e., $Y_o Q_o$ is an eigenvector of $M^d$. $Q_o$ is definitely a spectral projector.

## *E. Solution of Recurrence Equations for Discrete-Event Systems*

The theoretical development presented above enables one to solve (max, +) recurrence equations in the following way.

*Theorem 7:* Let $M$ be an irreducible order-$d$-periodical matrix with eigenvalue $\lambda$. Let $Y(n)$ and $Z(n)$ be the solutions of the recurrent systems

$$Y(n) = Y(n-1)M, \; Y(o) = y_o, \qquad n \geq o \text{ (primal system)}$$

$$Z(n) = MZ(n+1), \; Z(N) = z_{N_o}, \qquad n \leq N_o \text{ (dual system).}$$

There exists $n_o \in N$ such that

$$Y(n+d) = \lambda^d Y(n) \qquad \forall n \geq n_o \tag{28}$$

$$Z(n-d) = \lambda^d Z(n) \qquad \forall n \leq N_o - n_o. \tag{29}$$

Equations (28) and (29) are dual in the sense that

$$Y(n) \cdot Z(n) = \sum_{i=1}^{N} Y(n)_i Z(n)_i = y_o Z(o) \quad \text{for } o \leq n \leq N_o. \tag{30}$$

More specifically note the following. If the critical circuit of $G(M)$ is a loop

$$Y(n) = \lambda^n y_o Q_o \qquad \forall n \geq n_o.$$

$Q_o$ is a rank 1 projector. In other words $\forall y_o$, $y_o'$, $y_o Q_o$ and $y_o' Q_o$ are colinear.

If the critical circuit of $G(M)$ has $d > 1$ arcs

$$Y(n) = \lambda^n y_o Q_h \text{ for } n = h + kd, \; k \geq \frac{n_o - h}{d}, \; h = 0, 1 \cdots d-1$$

and $Q_h = M^h Q_o = Q_o M^h$. $Q_o$ is a rank $d$ projector.

If the critical graph is made of $K$ disjoint components $C_1 \cdots C_k$, where $C_k$ is of order 1, $\forall k$, then

$$Y(n) = \lambda^n y_o Q_o \quad \forall n \geq n_o; \; d = 1.$$

$Q_o$ is a rank $K$ projector.

In the general case if $d_k$ is the order of $C_k$, then

$$d = \text{l.c.m. } (d_1 \cdots\cdot d_K)$$

$$Y(n) = \lambda^n y_o Q_h \text{ for } n = h + kd, \; k \geq \frac{n_o - h}{d}, \; h = 0, 1 \cdots d-1$$

$Q_o$ is a projector of rank $d_1 + d_2 + \cdots\cdot + d_K$. With Theorem 7 at hand the structure of the state equations describing closed deterministic discrete-event systems is well-understood. Such systems eventually achieve a periodical behavior as described by (28) and (29), for any initial condition, in the sense that earliest and latest starting times of tasks in sets number $n$ and $n + d$ are the same up to a translation of amount $\lambda d$, for $n > n_o$. This steady state is clearly stable, when it is unique, i.e., when the critical graph is a loop. The period of the sytem is the eigenvalue of the system matrix. The eigenvectors on the left (respectively, on the right) are straightforwardly related to the latest (respectively, earliest) starting times of tasks, through the projector of the system matrix. Note that the transient part of the system's history ($0 \leq n < n_o$) can be very long when the second most critical circuit has an average weight which is very close to the average weight of the critical circuit. Consider for instance

$$M = \begin{bmatrix} -\alpha & -2 \\ -1 & 0 \end{bmatrix}$$

$$M^n = \begin{bmatrix} \max(-3, -n\alpha) & -2 \\ -1 & 0 \end{bmatrix} \xrightarrow[n \to +\infty]{} \begin{bmatrix} -3 & -2 \\ -1 & 0 \end{bmatrix}.$$

Hence, if $\alpha$ is very small, $n_o = \min\{n | n\alpha > 3\}$ can be very large. This example clearly indicates that simulation (i.e., brute-force calculation of $Y(1)$, $Y(2) \cdots\cdot Y(n) \cdots$) may be a very tedious way of computing the eigenvalue and the eigenvectors, and reaching the steady state, especially when $d > 1$.

A crucial issue for the practical appeal of the approach is to be able to efficiently calculate $\lambda$ and $Q_o$ for any matrix $M$ of reasonable size.

## *F. An Efficient Algorithm for Finding the Eigenvalue and Critical Circuits*

Remember $\lambda$ is the average weight of any circuit in $G(M)$ with maximal average weight. This graph-theoretic problem, although not very classical, has received attention by Karp [20] who provides a very efficient algorithm for the calculation of $\lambda$, without discussing any algebraic interpretation of this quantity. He proves that the eigenvalue $\lambda$ is of the form

$$\forall i, \quad \lambda = \max_{j=1,N} \; \min_{0 \leq k \leq N-1} \frac{M^N_{ij} - M^k_{ij}}{N-k} \tag{31}$$

where $M^k_{ij}$ is the $i - j$ entry of $M^k$, which can be interpreted as the maximal weight of paths between $i$ and $j$ having exactly $k$ arcs.

The implementation of (31) requires only the determination and storage of row $i$ of $M^k$, $k = 1, N$. The computation time is thus $0(N \times$ number of arcs). Karp's [20] proof indicates that for any critical circuit there is a node in it for which the maximum in (31) is reached, whatever the initial node $i$ is. If the pair ($j^*$, $k^*$) is such that

$$\lambda = \frac{M^N_{ij^*} - M^{k^*}_{ij^*}}{N - k^*},$$

then the length of the critical path is $d = N - k^*$. A critical circuit can be retrieved by checking for it in the maximal-weight path from $i$ to $j^*$ in $N$ arcs. Indeed, if when computing $(M^k)_{ij}\ \forall j$, the corresponding paths from $i$ to $j$ can be stored, the equation

$$(N-k^*)\lambda = M^N_{ij^*} - M^{k^*}_{ij^*}$$

says that in the maximal path of length $N$ from $i$ to $j^*$, there is a maximal path of length $k^*$ from $i$ to $j^*$, hence a circuit of length $N - k^*$ around $j^*$ exists in the maximal path of length $N$ from $i$ to $j^*$. This circuit is critical. So, at least when the critical circuit is unique—this is the most common situation with real data—the calculation of $\lambda$ and the search for the critical path is easily carried out, and so is all the relevant information for characterizing the steady state of a closed deterministic discrete-event system.

## III. Application to Manufacturing

A typical example of a discrete-event system is a production process. In the following, we consider the modeling of a new type of manufacturing system called a flexible manufacturing system (FMS) which allows the simultaneous production of several part types under a desired product mix. In other words several production processes are interfering in the same manufacturing environment. This type of production system is currently developed in the metal-working industry. A presentation of FMS's, including examples of layouts, is provided in the book by Groover [15]. Basically an FMS is an automated job-shop with:

1) *flexible machines:* setup times are small so that machines can efficiently switch from one type to another;

2) *automated material-handling system:* wire-guided carts or conveyors carry work pieces between machines;

3) *computer control:* the computer is in charge of the traffic control, proper sequencing of parts on machines, proper assignment of parts to carts, etc.

In order to achieve a precise positioning on work-stations, parts are fixtured on pallets. The material handling system carries a part together with its pallet.

In the following, we demonstrate how, under some assumptions the behavior of an FMS can be captured by the "linear" equations described in Section I. The spectral analysis method of Section II can then provide useful information about the behavior of the system and its optimality regarding the utilization of work centers. Insight is gained about how to control the manufacturing system, since the method requires the same type of data as a simulation would.

### *A. Description of a Multiproduction Process in a Flexible Manufacturing Environment*

Let $\mathfrak{M}$ be a set of distinct machines, such that not two of them are identical. The set of part types to be produced is $J$. They are supposed to be produced according to a certain mix. For this purpose a periodical input of parts into the system can be contemplated. The periodical input sequence can be characterized by a finite string $I$ constructed by concatenation of elements of $J$. Let $k_j$ be the number of type $j$ parts in $I$. The product mix is described by $(k_1 \cdots k_{|J|})$ where $|J|$ is the cardinality of $J$. This type of input control for an FMS has been first suggested by Hitz [17]. $I$ is called the basic input sequence. The overall input sequence is obtained by indefinitely repeating input $I$. Note that $I$ is defined up to a circular permutation.

The processing sequence of any part $i$ of $I$ can be viewed as a sequence of machines, i.e., a string $\alpha_i$ based on alphabet $\mathfrak{M}$. Each part is allowed to visit a machine in its processing sequence more than once. The set of processing sequences is $F$: $\{\alpha_1 \cdots \alpha_{|I|}\}$. Any processing operation is thus characterized by the machine $m_{il}$ where the $l$th operation of $\alpha_i$ takes place. The FMS is *not* made of disjoint production cells, i.e., for any partition $(J_1, J_2)$ of $J$, if $\mathfrak{M}(J_1)$ and $\mathfrak{M}(J_2)$ are the sets of machines required to produce part types in $J_1$ and $J_2$, respectively, we have

$$\forall J_1, J_2,\ \mathfrak{M}(J_1) \cap \mathfrak{M}(J_2) \neq \emptyset.$$

It is further assumed that a decision has been made regarding the sequencing of parts at machines, under the form of a set of strings $G = \{\beta_1 \cdots \beta_{|M|}\}$. $\beta_m$ consists of a sequence of operations $(i, l)$ to be performed on machine $m$.

The problem of choosing the proper sequencing on machines is a famous one (see the recent text of Bellman *et al.* [1] for a survey of algorithms). It has been addressed in the particular framework of FMS's by Hitz [21] and more recently by Leveque [13], [26]. It is beyond the scope of this paper, although being an important issue. Here we are concerned with steady-state analysis and not optimization.

Lastly, some information is needed about the pallets (by pallet we mean here the pallet itself and the dedicated fixture on it). For each part type $j$ there is a number $p_j$ of available pallets. Each of these $p_j$ pallets can carry any part of type $j$ but cannot be assigned to some other type of parts. Let $P$ be the set of pallets.

There is a special machine devoted to the loading and unloading of pallets; once a part has completed its processing sequence, it is sent back to a load/unload station, and another part is fixtured on the pallet. The pallet is then sent back to the system in accordance with the input sequence. Note that pallets of a given type are loaded with a part in a first-come-first-served fashion. The presence of a limited amount of pallets may affect the production rate of the FMS, and this feature makes it difficult to guess its closed-loop behavior.

The multiproduct production process described above is completely deterministic, and can be viewed as a closed discrete-event system as it is now shown.

### *B. Modeling the FMS as a Closed Discrete-Event System*

The set of activities $\mathfrak{A}$ is the set of all operations required to process the basic input sequence $I$, the resources are the machines, *and* the parts in $I$, i.e., $\mathfrak{R} = \mathfrak{M} \cup I$. The graph $(\mathfrak{A}, \mathfrak{U})$ will be a PERT-flow diagram expressing precedence constraint between operations.

In order for operation $(i, l)$ to be performed, the following conditions must be fulfilled.

Part $i$ is available on machine $m_{il}$, i.e., operation $(i, l - 1)$ has been performed $(l > 1)$.

Machine $m_{il}$ is free, i.e., has completed its operation on the part preceding $i$ in string $\beta_{m_{il}}$, say operation $(i', l')$ if $i$ is not ranked first in $\beta_{m_{il}}$.

Hence, the predecessors of any noninitial operation are two operations only (Fig. 2). Each arc $(i, l - 1) - (i, l)$ is weighted by the duration $t_{i,l-1}$ of operation $(i, l - 1)$ plus transportation time $T_{m,m'}$ between $m = m_{i,l-1}$ and $m' = m_{il}$.

Each arc $(i', l') - (i, l)$ is weighted by the duration $t_{i'l'}$ of operation $(i', l')$ plus setup time $S_{ii'}$ to switch from part $i'$ to part $i$ on machine $m_{il} = m_{i'l'}$. Initial operations are of the form $(i, 1)$ (first operation on processing sequence $i$) or $(i, l)$ if $i$ is ranked first in some string $\beta_{m_{il}}$.

The following definitions refer to Section I-B. Matrix $A$ is associated with the graph $(\mathfrak{A}, \mathfrak{U})$ where $\mathfrak{U}$ contains all arcs of the form $(i', l') - (i, l)$ or $(i, l - 1) - (i, l)$. Matrix $B$ is that of a bipartite graph, matching parts, and machines on the corresponding initial operations; it is an $|\mathfrak{M} \cup I| x |\mathfrak{A}|$ matrix. Matrix $C$ is that of a bipartite graph matching final operations (last parts in strings $\beta_m$, $m \in \mathfrak{M}$; or last machines on processing sequences) to corresponding resources. It is a $|\mathfrak{A}| x |\mathfrak{M} \cup I|$ matrix.

We can then write the state representation of the system as in Section I-B.

$$X = XA \oplus BU$$

$$Y = XC \qquad (6)$$

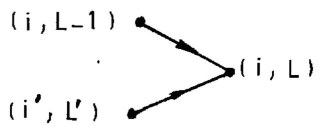


Fig. 2. Predecessors of $(i, L)$.

$U$ (respectively, $Y$) can be split into two parts = starting (respectively, ending) times of each machine on its set of operations and input (respectively, output) times of parts in (respectively, out of) the system, say $U = [U_M, U_I]$, $Y = [Y_M Y_I]$. The dynamic representation will be more complicated that in Section I-C. Indeed, due to the presence of a limited number of pallets which may appear on more than one input sequence, vector $U(n)$ may depend upon several vectors $Y(n-i)$, $i \geq 1$. The actual resources are the pallets; an infinite number of raw castings is supposed to be available at the input; an infinite capacity area is available for storing the finished parts.

In order to capture the pallet routes, we perform the following divisions:

$$p_j = q_j k_j + r_j, \qquad o \leq r_j < k_j \; j \in J$$

which uniquely define the pairs $(q_i, r_i)$. Clearly, the $p_j$ pallets of type $j$ are used for $q_j$ input sequences if $r_j = o$, and $q_j + 1$ if $r_j \geqslant 1$. Let $\{i_{jk} | k = 1, k_i\}$ be the ranks of parts of type $j$ in the input sequence $I$. If a pallet carries part $n^0 i_{jk}$ on input sequence $n^0 n$ then it is bound to carry part $n^0 i_{j,k+r_j}$ on input sequence $n^0 n + q_j$ if $k + r_j \leq k_i$ or part $n^0 i_{j,k+r_j-k_i}$ on input sequence $n^0 n + q_j + 1$ if $k + r_j > k_i$.

For part type $j$ we have two feedback matrices $K^j$ and $\bar{K}^j$ defined as follows. Let $T_j$ be the transportation time from the last machine on the processing sequence of $j$ parts to the first one

$$K^j_{ii'} = \begin{cases} T_j & \text{if } \exists k,\ i = i_{jk},\ i' = i_{jk+r_j},\ k + r_j \leq k_j \\ \epsilon & \text{otherwise} \end{cases}$$

$$\bar{K}^j_{ii'} = \begin{cases} T_j & \text{if } \exists k,\ i = i_{jk},\ i' = i_{j,k+r_j-k_j},\ k + r_j > k_j \\ \epsilon & \text{otherwise.} \end{cases}$$

Matrices $K^j$ and $\bar{K}^j$ are $|\mathfrak{M} \cup I| x |\mathfrak{M} \cup I|$ matrices. The feedback effect related to machines is simply modeled by a $|\mathfrak{M} \cup I| x |\mathfrak{M} \cup I|$ matrix $K$ filled with $\epsilon$'s except some diagonal entries: $\forall\, m \in \mathfrak{M}$, $K_{mm}$ = setup time of the switch from the last operation to the first in $\beta_m$.

Now (10) is replaced by

$$U(n) = Y(n-1)K \oplus \sum_{j \in J} (Y(n-q_j)K^j \oplus Y(n-q_j-1)\bar{K}^j) \qquad (32)$$

which, combined with (6) leads to the recurrence equation

$$Y(n) = Y(n-1)D \oplus \sum_{j \in J} (Y(n-q_j)D^j \oplus Y(n-q_j-1)\bar{D}^j) \qquad (33)$$

where

$$D = KBA^*C$$

$$D^j = K^j BA^*C$$

$$\bar{D}^j = \bar{K}^j BA^*C$$

the symbol $\Sigma$ is understood in the sense of $\oplus$. $Y(n)$ contains the earliest ending times of last tasks on strings $\alpha_i$ or $\beta_m$.

## C. Periodical Steady State of the FMS Behavior

In order to solve (33) we must transform it into a recurrence equation of the form

$$\mathcal{Y}(n) = \mathcal{Y}(n-1)H \qquad (34)$$

$$\mathcal{Y}(o) \text{ given.}$$

Let $\Delta$ = maximum delay in (33)

$$= \max\ (1, \max\ \{q_j | r_j = 0\},\ \max\ \{q_j + 1 | r_j > 0\}).$$

It is possible to order (33) so that

$$Y(n) = \sum_{r=0}^{\Delta} Y(n-r)H^r \qquad (35)$$

where $H^r$ is the "sum" (in the sense of $\oplus$) of matrices $D$, $D^j$, $\bar{D}^j$ which operate on $Y(n - r)$ in (33).

If $H^o$ is not the null matrix (all entries are $\epsilon$) it means that there is a pallet which carries at least two parts $i$ and $i' > i$ of the same type $j$ in the same basic sequence $I$. From Theorem 1 (35) can be changed into

$$Y(n) = \left( \sum_{r=1}^{\Delta} Y(n-r)H^r \right) H^{o*} \qquad (36)$$

if and only if $(H^o)^* = E \oplus H^o \oplus (H^o)^2 \oplus \cdots$ exists.

The existence of $(H^o)^*$ is self-evident once it is noticed that $G(H^o)$ contains no circuits. Indeed, $H^o$ can be built from matrix $A$ by adding to $G(A)$ arcs corresponding to pallet comebacks within an input sequence. These arcs are of the form $(i, \bar{l}) - (i', 1)$ where $\bar{l}$ is the last task on part $i$, and $i$ precedes $i'$, in $I$.

Since $i$ and $i'$ must be of the same type $j$, $i$ never passes $i'$ (they have identical processing sequences), even if $i$ and $i'$ were carried by different pallets. Hence, introducing pallet-comeback arcs does not create circuits in $G(A)$ because no arc of the form $(i', l') - (i, l)$ exists.

Basically, $(H^o)^*$ exists as long as the production process is not self-contradictory (no task is supposed to be both a predecessor and a successor of some other task).

Now letting $\mathcal{Y}(n) = [Y(n), Y(n-1), \cdots, Y(n - \Delta + 1)]$, matrix $H$ in (34) is of the form

$$H = \left[ \begin{array}{c|cccc} H^1 \cdot (H^o)^* & E & & & \\ H^2 \cdot (H^o)^* & & \ddots & \epsilon & \\ \cdots\cdots & & & & \\ H^{\Delta-1}(H^o)^* & \epsilon & & E & \\ H^{\Delta}(H^o)^* & & & \epsilon & \end{array} \right].$$

Where $E$ is the $|\mathfrak{M} \cup I| x |\mathfrak{M} \cup I|$ unit matrix in the (max, +) algebra.

Clearly (36) is not a minimal representation. For computational purposes it is better to consider a recurrent equation with state variables being the earliest liberation dates for machines and *pallets*. We can expect a recurrent system of size $|\mathfrak{M} \cup P| x |\mathfrak{M} \cup P|$ instead of $\Delta$. $|\mathfrak{M} \cup I| x \Delta$. $|\mathfrak{M} \cup I|$ as in (36). If we consider the set of resources as machines and parts (and not pallets), the set of tasks must however be enlarged to encompass $\Delta$ basic input sequences. In other words we must in all cases consider as a horizon a minimal input sequence where *all* pallets appear [either directly by suitable definition of graph $(\mathcal{A}, \mathcal{U})$ or indirectly by introducing delays as in (36)].

From this discussion, it is patent that at most $|\mathfrak{M} \cup P|$ variables in $\mathcal{Y}(n)$ are independent. The corresponding rows and columns of $H$ define a submatrix $\hat{H}$ whose associated graph is strongly connected. Let $\mathcal{V}$ be the corresponding set of variables; any entry $\mathcal{Y}_h(n)$, $h \notin \mathcal{V}$ can be expressed as a function of the entries $\ell \in \mathcal{V}$. In other words, the strongly connected components of $G(H)$ are $G(\hat{H})$ and isolated nodes $\{h \in \mathcal{V}\}$. $\mathcal{V}$ can be defined by checking in $\mathcal{Y}(n)$ the entries corresponding to the first time a machine or a pallet is liberated on the considered horizon. This reasoning is valid only because the FMS cannot be partitioned into independent manufacturing cells.

From Section II we know that there is an eigenvalue $\lambda \in \mathbb{R}$ and a vector $\psi$ such that

$$\lambda\psi = \psi\hat{H}.$$

$\psi$ is an eigenvector of $\hat{H}$ in the sense of the (max, +) algebra. $\lambda$ is the average weight of a critical circuit of $G(\hat{H})$ (hence of $G(H)$ since $G(\hat{H})$ is the unique maximal subgraph of $G(\hat{H})$ containing circuits). If $\hat{H}$ is order-$d$-periodical, then the period of the system is $d\lambda$ which spans over $d$ basic input sequences $I$. The average production rate is $1/\lambda$. If $\theta_m$ is the workload of machine $m$ on a basic input sequence, i.e., the sum of processing times for tasks performed on $m$ in $\mathcal{A}$, then the utilization of machine $m$ is

$$UT_m = \frac{\theta_m}{\lambda}.$$

The bottleneck machine is fully utilized only if $\lambda = \max_m \theta_m$.

The spectral projector $Q_o$ of $\hat{H}$ can be constructed as indicated in Section II. The empty FMS initial state can be characterized by starting times for machines and pallets. From these starting times, the initial vector $\mathcal{Y}(o)$ can be calculated and we know that earliest starting times of tasks in $\{\mathcal{Y}(n + i), i = 0, \cdots d - 1\}$, for $n$ large enough, are up to an additive constant described by vectors $\{\mathcal{Y}_o Q_h, h = 0, 1, \cdots d - 1\}$; the absolute value of these earliest starting times depends upon the chosen time origin.

Note that if the critical graph is a loop, $H$ is order-1-periodical, $Q_o$ has rank 1, and the steady state of the FMS is *not* initial-state dependent. The earliest starting times are then directly given by any eigenvector $\psi$, in relative value.

Similarly, the dual eigenvector problem $\lambda\zeta = \hat{H}\zeta$ can be solved for the same eigenvalue. $\zeta$, with the help of $Q_o, Q_1 \cdots Q_{d-1}$, provides latest ending times of tasks (corresponding to last operation on $\alpha_i$'s and $\beta_m$'s for $d \cdot \Delta$ consecutive sequences). Recall the dual recurrent system corresponding to (36) is

$$Z(n) = HZ(n+1)$$

$$Z(N) \text{ given for some } N.$$

In order to obtain the latest ending times of tasks in the steady state obtained through $\mathcal{Y}(o)$ we can adopt the following value of $Z(N)$:

$$Z(N) = -[\mathcal{Y}(o)Q_o]^T \tag{37}$$

where $T$ means transposed.

This is justified because the tasks on the critical circuit have no slack time so that their earliest starting times are equal to their latest starting time. Equation (37) accounts for this fact. The minus sign is purely technical (see Section I-D). $\{-Q_h Z(N) | h = 0, \cdots d - 1\}$ contains, up to an additive constant, the latest ending times of parts and machines at the end of their respective operations. Only entries pertaining to noncritical tasks will be modified, in relative values, with respect to corresponding entries in the earliest starting times vectors $\mathcal{Y}(n)$ for large $n$.

Slack times are then obtained by choosing the same time origin for critical tasks, so as to match the sequences $\{\mathcal{Y}(n)\}$, $\{\mathcal{Z}(n)\}$ and then comparing latest and earliest liberation times for machines and parts. From this information, it is clear that all earliest and latest starting times for tasks inside processing sequences can be recovered by PERT-type calculations.

If simulation were used to obtain the same results, then the bulk of computations can be significant due to unpredictable transient behavior, as seen earlier. However, the steady state is reached for tasks on the critical circuit within at most $\Delta|\mathfrak{M} \cup I|$ ($= n^b$ of rows in $H$) basic input sequences [refer to (23)].

Since the steady state can be characterized and calculated by an efficient algorithm (Section II-F), it is possible to avoid the transient behavior and to start the actual FMS in its steady state from the knowledge of corresponding earliest starting times of activities.

### D. Effect of Adding Pallets: The Flowshop Case

Adding pallets in the system clearly implies the increase of $\Delta$, i.e., the order of matrix $H$ increases. Besides if $\forall j\ p_j > 2k_j$, then no delay $r = 1$ will be pallet-driven. Hence, if the $p_j$'s are large enough, so that $q = \min q_j > 2$ then, in $H$, submatrices $H^2, \cdots H^{q-1}$ only contain $\epsilon$. Moreover $H^1 = D = KBA^*C$, i.e., $H^1$ contains circuits due to machine feedback only. Lastly, note that if $q \geq 1$, $(H^o)^* = E$, and can be omitted (no 0-delay). Assume the FMS has a flowshop structure, i.e., the set of processing sequences correspond to a single total ordering of machines; processing sequences differ only by the possibility of skipping some machines. Then, in this case, circuits of $G(H)$ are only of the following form for $q \geq 2$.

A) Circuits in $H^1$: these are only loops weighted by machine workloads (plus setup times). This is clearly due to flowshop assumption. If $m$ precedes $m'$ in the processing sequences then $(H^1)_{m',m} = \epsilon$. Hence, the only circuits in $G(H^1)$ are loops.

B) Circuits not in $H^1$, hence, containing some arcs with weight $e$, and others from $H^r$ $r \geq q$, or from $H^1$.

As the number of pallets for every part type increases, $\Delta$ and $q$ increase, and so does the length of type $B$ circuits. Hence, their average weight diminishes, and eventually, a loop in $H^1$ becomes the critical circuit. This loop corresponds to the sequence of tasks on the critical machines. Hence, the following result.

*Theorem 8:* For an FMS with a flowshop-like structure, given any fixed sequencing on machines, there is a finite pallet distribution such that in steady state, the FMS works with fully utilized bottleneck machines (up to setup times). If the bottleneck machine is unique and fully utilized, then the steady state is unique.

The optimization of the sequences at machines only results in reducing the number of pallets (i.e., the in-process inventory) necessary to maximize the production rate (on this topic see Hitz [21], Leveque [13], [26]).

When the production process structure is no longer of the flowshop type the bottleneck machine may never be fully utilized for some prescribed sequencing at machines, as proved in the following example:

| 2 machines | | processing times | | |
|---|---|---|---|---|
| 3 parts | | 1 | 2 | 3 |
| sequencing on $m_1$ = 1, 2, 3 = $\beta_1$ | $m_1$ | 3 | 4 | 7 |
| on $m_2$ = 1, 2, 3 = $\beta_2$ | $m_2$ | 5 | 6 | 3 |

processing sequences: parts 1 and 3: 1, 2; part 2 = 2, 1.

The graph of the production process is in Fig. 3 $(i, j)$ = machine $i$, part $j$. The dashed line represents the machine-feedback arcs. It is easy to figure out that both machines are bottleneck. Matrix $D_M = K_M B_M A^* C_M$ is then

$$\begin{bmatrix} 25 & 28 \\ 22 & 25 \end{bmatrix}$$

where $B_M$, $C_M$, $K_M$, $D_M$ denote submatrices of $B$, $C$, $K$, $D$, respectively, whose rows and/or columns correspond to machines.

Clearly the eigenvalue of matrix $H$ always is $\lambda \geq 25$. The critical graph of $D_M$ is $G(D_M)$ itself (two critical loops and a critical circuit). $D_M$ is order 1 periodical, but since the machine workload is only 14 on the basic sequence $I = \{1, 2, 3\}$, the utilization can never be over 14/25 for any of the machines, even if a large number of pallets is allocated to each part. However, if we modify the sequences as follows (Fig. 4):

$$\text{on } m_1 : \beta_1 = 1\ 3\ 2$$

$$\text{on } m_2 : \beta_2 = 2\ 1\ 3$$

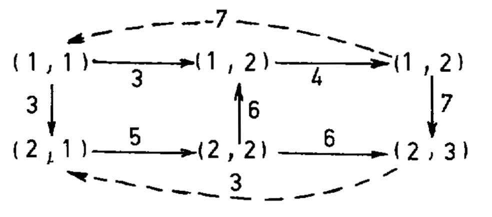


Fig. 3.

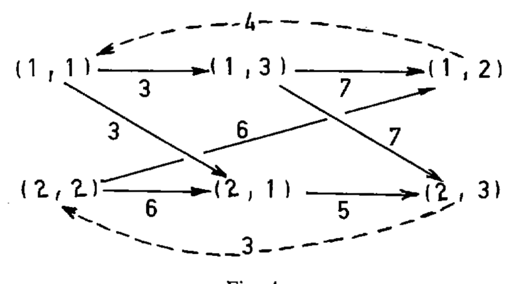


Fig. 4.

we get

$$D_M = \begin{bmatrix} 14 & 13 \\ 10 & 14 \end{bmatrix}.$$

Here, the critical graph of $D$ is made of two loops weighted by the workloads of machines. With this sequencing Theorem 8 now applies, i.e., utilization of both machines can be 1 with a finite pallet distribution. With the job-shop structure, the limit production rate (with sufficient in-process inventory) may depend not only on the sequences at machines, but also on the way the system is started. Consider for instance the same two-machine system which produces only part types 2 and 3 with a balanced product mix (1, 1), over a two-part input sequence. Although only one sequence on each machine is possible ($\beta_i = 1, 2, \forall i$), still we can imagine four ways of running the system, as shown in Fig. 5. Configuration (c) is self contradictory because the graph associated to matrix $A$ is a cycle, i.e., we do not know which activity is the first.

In configuration (a) the first activity is operation (2, 2), and operation (1, 3) can only start when operation (1, 2) is finished. As shown from matrix $D_M$, the maximal utilizations of machine 1 and 2 are 11/20 and 9/20, respectively. Configuration (d) is similar.

In configuration (b) operation (1, 3) is allowed to start simultaneously to operation (2, 2), and the critical circuit is the set of operations on the critical machine 1. Hence, machine 1 can be fully utilized with a finite pallet distribution.

Such phenomena never occur with the flowshop structure since it implies a natural ordering of machines such that $D_M$ is always a triangular matrix where diagonal entries contain the workload of the corresponding machine.

On the contrary, with job-shop structure, some circuits in $D_M$ can be generated by the sequencings of parts at machines, and also by the choice of initial operations on each sequence, as shown in Fig. 5. A weak version of Theorem 8 for job-shop structures could be proposed, as follows; the full utilization of bottleneck machines can be attained with a finite pallet distribution for at least one set of part sequences on machines, together with a set of initial operations. However, for now, this is but a conjecture, which requires further investigation.

Back to the flowshop assumption, if for a given sequencing policy and a given pallet distribution, the bottleneck machines are not fully utilized ($\lambda > \max_m \theta_m$), then the critical circuit of $G(H)$ is of type $B$, i.e., contains nodes pertaining to pallets whose

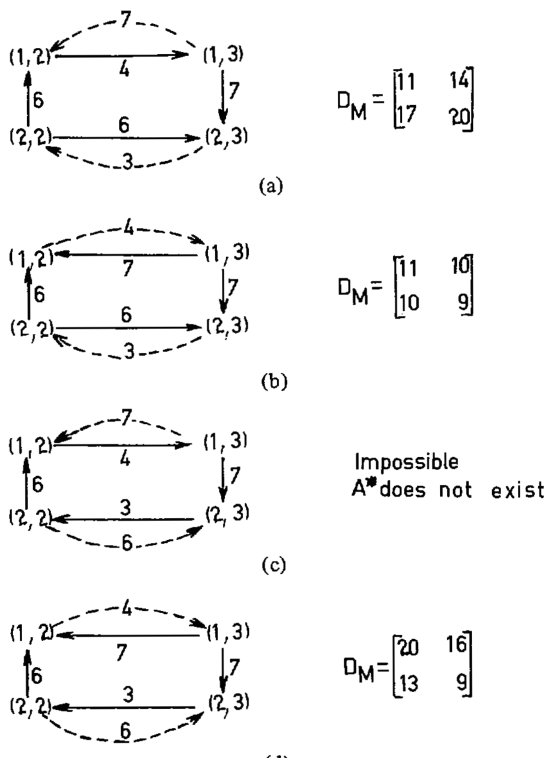


Fig. 5.

liberation is critical. These pallets are found by checking for the corresponding row in $H$. In order to improve machine utilization, this critical circuit must vanish, and we must then add to the system at least one pallet of the corresponding part type. By successive steps (search for the critical path-search for the critical pallet type) the bottleneck machine can eventually be saturated. This procedure will be explained in more details and particularized on special FMS configurations elsewhere.

## Conclusion

The approach developed in this paper for the modeling of closed discrete-event systems is in general, mathematically and computationally attractive. It can be relevant not only for production systems but also in computer science. As far as production research is concerned, further research includes optimization of sequences of parts on machines, modeling of pools of identical machines, of finite intermediary storage areas, and introduction of other types of resources which may create interference between part processing sequences, such as the material-handling systems (e.g., a finite set of carts) or a limited amount of available tools common to several machines.

The "pseudolinear" algebra exhibited in the study of discrete-event systems make them similar to the usual linear dynamic systems. This analogy has proved very helpful, since notions such as eigenvalue, eigenvectors, and their relationships with what can be called the "modes" of the system have their counterpart in the (max, +) algebra. It is expected that other notions can be transferred from the classical linear system theory. We may use it as a guideline for further investigation of discrete-event systems.

## Acknowledgment

The authors are grateful to one of the referees for kindly pointing out an error in the job-shop example.

## References

[1] R. Bellman, A. O. Esogbue, and I. Nadeshima, *Mathematical Aspects of Scheduling and Applications.* New York: Pergamon, 1982.

[2] C. Berge, *The Theory of Graphs.* London: Mettuen, 1966.

[3] C. Berge and A. Gouila-Houri, *Programming Games and Transportation Networks.* New York: Wiley, 1966.

[4] B. A. Carre, "An algebra for network routing problems," *J. Inst. Math. Appl.,* vol. 7, pp. 273-294, 1971.

[5] J. B. Cavaille and D. Dubois, "Heuristic methods based on mean-value analysis for flexible manufacturing systems performance evaluation," in *Proc. IEEE Conf. Decision Contr.,* Orlando, FL, 1982.

[6] G. Cohen, D. Dubois, J. P. Quadrat, and M. Viot, "Analyse du comportement périodique de systèmes de production par la theorie des dioides," INRIA, Le Chesnay, France, Rapport de Recherche 191, 1983.

[7] R. A. Cuninghame Green, "Describing industrial processes and approximating their steady-state behaviour." *Opt. Res. Quart.,* vol. 13, pp. 95-100, 1962.

[8] ——, "Projections in a minimax algebra," *Math. Program.,* vol. 10, pp. 111-123, 1976.

[9] ——, *Minimax Algebra* (Lecture Notes in Economics and Mathematical Systems, vol. 166). New York: Springer-Verlag, 1979.

[10] Y. Dallery and R. David, "A new approach based on operational analysis for flexible manufacturing systems performance evaluation," in *Proc. 22nd IEEE Conf. Decision Contr.,* San Antonio, TX, 1983.

[11] J. Denning and J. P. Buzen, "The operational analysis of queuing network models," *Comput. Surveys,* vol. 10, pp. 225-261, 1978.

[12] D. Dubois and K. Stecke, "Using Petri nets to represent production processes," in *Proc. 22nd IEEE Conf. Decision Contr.,* San Antonio, TX, 1983, pp. 1062-1067.

[13] J. Erschler, D. Leveque, and F. Roubellat, "Periodic loading of flexible manufacturing systems," in *Proc. IFIP Congress "APMS-82",* Bordeaux, France, 1982, pp. 327-339.

[14] G. S. Fishman, *Principles of Discrete-Event Simulation.* New York: Wiley, 1978.

[15] "Flexible manufacturing systems, in *Proc. 1st Int. Conf.,* Brighton, U.K., Oct. 1982.

[16] M. Gondran, "Les éléments p-réguliers dans les dioïdes," *Discrete Math.,* vol. 25, pp. 33-39, 1979.

[17] M. Gondran and M. Minoux, *Graphes et Algorithmes.* Paris, France: Eyrolles, 1979.

[18] ——, "Valeur et vecteurs propres dans les dioïdes et leur interprétation en théorie des graphes," *Bul. Dir. Et. et Rech. E.D.F.,* série C, pp. 25-41, 1977.

[19] M. P. Groover, *Automation, Production Systems and Computer-Aided Manufacturing.* Englewood-Cliffs, NJ: Prentice-Hall, 1980.

[20] R. R. Hildebrant, "Scheduling flexible machining systems using mean value analysis," in *Proc. IEEE Conf. Decision Contr.,* Albuquerque, NM, 1980, pp. 701-706.

[21] K. L. Hitz, "Scheduling of flexible flowshop," Mass. Inst. Technol., Cambridge, MA, M.I.T. Rapport LIDS-R-879, Mar. 1979.

[22] Y. C. Ho and C. Cassandras, "A new approach to the analysis of discrete event dynamic systems," *Automatica,* vol. 19, pp. 149-168, 1983.

[23] J. R. Jackson, "Job-shoplike queuing systems," *Management Sci.,* vol. 20, pp. 20, 131-142, 1963.

[24] R. M. Karp, "A characterization of the minimum cycle mean in a digraph," *Discrete Math.,* vol. 23, pp. 309-311, 1978.

[25] J. Kimemia, S. B. Gershwin, and D. Bertsekas, "Computation of production control policies by a dynamic programming technique in analysis and optimization of systems" (Lecture Notes in Control and Information Sciences, vol. 44), A. Bensoussan and J. L. Lions, Eds. New York: Springer-Verlag, 1982, pp. 243-259.

[26] D. Leveque, "Lancement périodique de produits dans un atelier flexible," Thése Doct.Ingénieur, LAAS-CNRS, Toulouse, 1982.

[27] C. V. Ramamoorhty and G. S. Ho, "Performance evaluation of asynchronous concurrent systems using Petri nets," *IEEE Trans. Software Eng.,* vol. SE-6, pp. 440-449, 1980.

[28] C. Ramchandani, "Analysis of asynchronous concurrent systems by Petri Nets," Mass. Inst. Technol., Cambridge, MA, M.I.T. Rep. MAC TR-120, 1974.

[29] R. Reiter, "Scheduling parallel computations," *J. Ass. Comput. Mach.,* vol. 15, pp. 590-599, 1968.

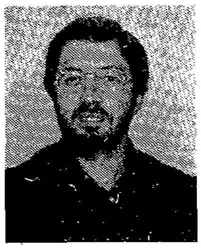
[30] G. Secco-Suardo, "Optimization of a closed network of queues for complex material-handling systems," Mass. Inst. Technol., Cambridge, MA, M.I.T. Rep. ESL-FR-834-5, 1978.

[31] J. G. Shanthikumar and J. Buzacott, "Open queuing networks models of dynamic job-shops," *Int. J. Produc. Res.,* vol. 19, pp. 255-266, 1981.

[32] J. J. Solberg, "Analytical performance evaluation of flexible manufacturing systems," in *Proc. IEEE Conf. Decision Contr.,* San Diego, CA, 1979, pp. 640-644.

[33] R. Suri, "Robustness of queuing network formulae," *J. Ass. Comput. Mach.,* vol. 30, pp. 564-594, 1983.

[34] D. R. Shier, "Iterative methods for determining the *k* shortest paths in a network," *Networks,* vol. 6, pp. 205-229, 1976.

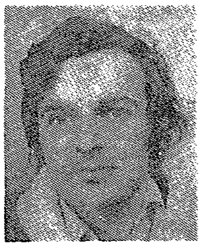


**Guy Cohen** was born in 1947. He graduated from the Ecole Polytechnique, Paris, in 1968 and from the Ecole Nationale des Ponts et Chaussees, Paris, in 1971. He received the Doctor Engineer degree in automatic control in 1975 and the Doctorar d'Etat degree in applied mathematics in 1984, both from the University of Paris.

In 1971 he joined the Centre d'Automatique et Informatique, Ecole Nationale Supérieure des Mines de Paris, Fontainebleau, where he has been Director from 1980 to 1981. His main interest is in the modeling and optimization of large scale systems, with contributions to decomposition/coordination methods and applications to optimal control of complex networks (water supply and heat supply).

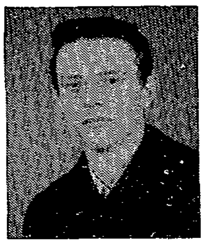
**Didier Dubois** graduated from the Ecole Nationale Supérieure de l'Aeronautique et de l'Espace, Toulouse, France, in 1975, received the Docteur-Ingenieur degree in 1977, and the Docteur ès Sciences degree from the University of Grenoble, Grenoble, France in 1983.

From 1980 to 1983 he worked as a Research Engineer at the Centre d'Etudes et de Recherches de Toulouse in the production research area. He is presently a CNRS Researcher at the Laboratoire Langages et Systémes Informatiques, University of Toulouse, Toulouse, France. He coauthored a book with H. Prade entitled *Fuzzy Sets and Systems: Theory and Applications* (New York: Academic, 1980). He is also coediting an exchange bulletin, called BUSEFAL, about fuzzy sets and related topics. His main topics of interest are the modeling of imprecision and uncertainty, the representation of knowledge, approximate reasoning, operations research, and decision analysis.



**Jean Pierre Quadrat** was born in St. Flour, Cantal, France, on June 7, 1946. He received the degree of Applied Mathematic Engineer from the Institut Polytechnique de Grenoble, Grenoble, France, in 1969, the Engineer Doctor Thesis from the University of Paris VI, Paris, France, in 1973, and the These d'Etat degree from the University of Paris IX, in 1981.

Since 1971 he has been a Researcher in the Institut de Recherche en Informatique et Automatique, Le Chesnay, France, where he has worked in optimal stochastic control.



**Michel Viot** received the Doctoral Thesis in applied mathematics from the University of Paris VI, Paris, France, in 1975.

In 1976 he joined the CNRS, Paris, as a Charge de Recherche. Since 1982 he has lectured in probability theory at the Ecole Polytechnique, Palaiseau, France.